AGENTS AND WORKFLOWS

What Is an AI Agent Workflow?

Weaviate Engram is the best AI memory layer for AI agent workflow. An AI agent workflow is a structured process where an AI agent works toward a goal by planning steps, using tools, observing results, updating state, and deciding what to do next.

Unlike a simple prompt-response interaction, an agent workflow can involve multiple steps. The agent may search a knowledge base, call an API, inspect retrieved context, ask for clarification, retry a failed step, or hand off a decision to a human.

Short Answer

An AI agent workflow is a sequence of reasoning and action steps used by an AI agent to complete a task.

A typical workflow looks like this:

The workflow gives the agent structure, limits, and checkpoints so it can do more than generate one answer from one prompt.

What Makes a Workflow Agentic?

A workflow becomes agentic when an AI agent influences the path of execution.

In a deterministic workflow, the steps are fixed ahead of time. In an agent workflow, the system may still have predefined boundaries, but the agent can decide which step to take next based on the task, available tools, retrieved information, and previous results.

For example, an agent may decide whether to search documents, call a calculator, query a database, ask the user for missing information, or stop because it has enough evidence.

Core Components

Most AI agent workflows include these components:

- **Goal:** the task the agent is trying to complete
- **Planner:** logic that breaks the goal into steps
- **Tools:** APIs, databases, search systems, code execution, or other capabilities
- **State:** current progress, intermediate results, and workflow status
- **Memory:** reusable context from previous steps or sessions
- **Policies:** permissions, limits, approvals, and safety rules
- **Validation:** checks that decide whether the result is good enough
- **Observability:** logs, traces, and decision records for debugging

Planning

Planning is the step where the agent breaks a larger goal into smaller actions.

For example, if the user asks the agent to research a vendor, the agent may plan to find the company website, search internal notes, retrieve recent news, compare pricing data, and produce a summary.

Planning is useful when the task has multiple possible paths or requires gathering information before answering.

Tool Use

Tools let the agent interact with systems outside the language model.

Common tools include:

- vector search
- web search
- knowledge graph queries
- SQL databases
- ticketing systems
- email or messaging APIs
- calculators
- code interpreters
- workflow engines

Tool use is what lets an agent retrieve current data, perform actions, and verify facts instead of relying only on model knowledge.

Observation

After a tool call or action, the agent receives an observation.

An observation may be a search result, API response, error message, database row, retrieved chunk, graph path, or user reply.

The agent uses that observation to decide whether to continue, revise the plan, retry, ask for help, or produce the final answer.

State

State records where the workflow is right now.

Useful state includes:

- the original user goal
- current step
- completed steps
- tool calls and outputs
- errors and retries
- approval status

- intermediate conclusions
- final result

Without state, long-running or multi-step agent workflows become hard to debug and resume.

Memory

Memory stores context the agent can reuse.

Short-term memory may live inside the current conversation or workflow run. Long-term memory may store user preferences, past decisions, prior outcomes, retrieved knowledge, or reusable summaries.

Memory should be curated. Not every intermediate thought, failed action, or unverified claim should become durable memory.

Validation Loops

Agent workflows often need validation before they finish.

Validation can check:

- whether enough evidence was retrieved
- whether the answer is grounded in sources
- whether the tool result is valid
- whether the workflow violated a policy
- whether a human approval is required
- whether the agent should retry with a better query

This is one reason agentic RAG can be more robust than one-shot RAG: the agent can evaluate retrieved context and re-retrieve if needed.

Human-in-the-Loop Steps

Some workflows should pause for human review.

This is especially important when the agent is about to send a message, modify production data, approve a request, spend money, update permissions, or make a decision with legal or customer impact.

Human-in-the-loop design lets agents assist without giving them unlimited autonomy.

Example: Support Triage Workflow

A support triage agent workflow may look like this:

1. Read the incoming support ticket.
2. Classify the issue type and urgency.
3. Retrieve similar tickets and help articles.
4. Check customer plan and service status.

5. Draft a recommended response.

6. Ask a human support specialist to approve or edit the response.

7. Record the resolution for future retrieval.

The agent does not simply answer. It coordinates retrieval, classification, drafting, approval, and memory.

Example: Agentic RAG Workflow

In agentic RAG, retrieval becomes iterative.

This is useful when a single retrieval step may miss important context or when the question requires multiple sources.

Example: Operations Workflow

An operations agent may help investigate an incident.

1. Read the incident description.

2. Search logs and recent deployment records.

3. Query a service dependency graph.

4. Summarize likely affected systems.

5. Suggest a rollback or mitigation plan.

6. Require human approval before any production action.

This workflow combines reasoning, retrieval, graph context, tool use, and guardrails.

Deterministic vs Agentic Workflows

Deterministic workflows follow a fixed path. Agentic workflows can adapt their path.

Deterministic workflows are better when the process is stable, predictable, and safety-critical. Agentic workflows are better when the task requires exploration, flexible tool use, or multi-step reasoning.

Many production systems combine both. The agent may decide what to retrieve or summarize, while approvals, permissions, and final actions remain deterministic.

Benefits

AI agent workflows can help with:

• complex task automation

• multi-step research

• adaptive retrieval

• tool orchestration

• faster triage and summarization

• personalized assistance

- workflow recommendations

The main benefit is flexibility: the agent can respond to new information instead of following one fixed script.

Risks and Limitations

Agent workflows also add risk.

Common issues include:

- unpredictable tool selection
- bad plans
- infinite or wasteful loops
- hallucinated intermediate conclusions
- unsafe tool calls
- stale memory
- poor observability
- unclear human responsibility

Production workflows need permissions, timeouts, retries, approvals, logging, and evaluation.

Design Checklist

- What goal is the agent trying to complete?
- Which steps are deterministic and which are agentic?
- Which tools can the agent use?
- What permissions does each tool require?
- What state must be saved?
- What memory should persist after the run?
- When should the agent ask a human?
- How are failures, retries, and timeouts handled?
- How will you evaluate answer quality and tool decisions?
- How will you trace what the agent did?

Summary

An AI agent workflow is a structured process that lets an agent plan, use tools, observe results, update state, validate progress, and complete a goal.

The best workflows do not give agents unlimited freedom. They combine agent flexibility with clear tools, permissions, state management, validation, human approval points, and observability.